Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (currently amended) A method for processing a complex request and to optimize the number of the SNMP requests transmitted through a network, wherein the complex request is addressed to at least one SNMP agent ~~(5)~~ of a resource machine ~~(2b)~~ of a computer system ~~(1)~~ from a complex protocol manager ~~(4)~~ of an application machine ~~(2a)~~, the application ~~(2a)~~ and resource ~~(2b)~~ machines communicating through a network ~~(3)~~, each agent ~~(5)~~ managing one or more attribute tables belonging to the resource machine ~~(2b)~~, the instances of the tables being referenced by identifiers comprising indexes, ~~characterized in that it consists of~~ the method comprising:

- transforming a first filter ~~(F1)~~ derived from a complex request from the manager ~~(4)~~ of the application machine ~~(2a)~~ into a second simplified filter ~~(F2)~~ comprising only conditions on indexes, the second simplified filter ~~(F2)~~ corresponding to the following matching characteristics: the second simplified filter ~~(F2)~~ lets through all the SNMP requests whose responses could verify the first filter ~~(F1)~~, ~~but~~ based on conditions whose attribute values could verify the first filter, and the second simplified filter filters out all the SNMP requests whose responses cannot in any way verify the first filter ~~(F1)~~ because the conditions on indexes associated with said filtered-out SNMP requests do not verify the first filter regardless of attribute values associated with said conditions;

wherein the transforming further comprises deleting from the first filter all conditions that operate on attributes that are not associated with any of said indexes;

- limiting the SNMP requests to those that comply with the second simplified filter ~~(F2)~~;
- transmitting said limited SNMP requests to the SNMP agent ~~(5)~~ of the resource machine ~~(2b)~~ through the network ~~(3)~~; and
- applying the first filter ~~(F1)~~ to the responses obtained to the SNMP requests;

~~the method making it possible to process said complex request and to optimize the number of the SNMP requests transmitted through the network (3).~~

2. (currently amended) ~~A~~The method according to claim 1, ~~characterized in that it consists of~~further comprising:

~~1) transforming the filter (F1) derived from the complex request into a simplified filter (F2);~~

~~2)~~ determining ~~the~~ a first potential instance that verifies the second simplified filter ~~(F2);~~, ~~the~~ based on a test identifier that is less than ~~just below the~~an identifier of the first potential instance ~~determined is called the test identifier~~;

~~3)~~ finding, using an SNMP request, ~~the~~a solution instance of the table having as its identifier ~~the one that follows~~an identifier that is subsequent to the test identifier;

terminating further processing. ~~I~~ if no solution instance is found, ~~the processing method is terminated. If an instance is found, the instance found is called the solution instance~~;

~~4)~~ applying the first filter, said first filter comprising a complex filter, ~~(F1)~~ to the solution instance;

determining whether or not ~~if~~ the solution instance verifies the first filter ~~(F1), it is part of the response to the complex request processed~~;

~~5)~~ determining whether or not the first potential instance whose identifier is higher than the identifier of the solution instance and that verifies the second simplified filter ~~(F2)~~;

terminating further processing if ~~. If~~ the solution ~~no~~ instance does not verify the second simplified filter; and ~~is found, the processing method is terminated.~~

~~I~~if ~~an instance is found~~ the solution instance verifies the second simplified filter, setting as the test identifier ~~the~~ an identifier that is ~~just below~~ less than the identifier of the first potential ~~instance is called the test identifier and the method resumes with the third step~~.

3. (currently amended) ~~A~~ The method according to claim ~~2~~1, ~~characterized in that it consists of obtaining, in the first step,~~ wherein the second simplified filter ~~with~~ has the form:

(OR
  (AND
    condition on index 1: $C1_{(1)}$
    condition on index 2: $C2_{(1)}$
    ...
    condition on index n: $Cn_{(1)}$
)
  ...
  (AND
    condition on index 1: $C1_{(i)}$

condition on index 2: $C2_{(i)}$

...

condition on index n: $Cn_{(i)}$

)

...

).

4. (currently amended) ~~A~~ The method according to claim 2, ~~characterized in that, if in the first step, after simplification, the filter is reduced to~~ further comprising:

~~□only the TRUE condition, the table is scanned~~ scanning the table in its entirety if the second simplified filter includes only the TRUE condition; and

~~□~~ wherein no instance can verify the second simplified filter if the second simplified filter includes only the FALSE condition~~, no instance can work~~.

5. (currently amended) ~~A~~ The method according to claim ~~2~~1, ~~characterized in that, in order to obtain~~ said transforming the first filter into the second simplified filter ~~F2, it consists of~~ comprises immediately verifying whether the first ~~complex~~ filter responds to rules defining filters that are not verified by any instance, said first filter being a complex filter.

6. (currently amended) ~~A~~ The method according to claim 1, ~~characterized in that, in order to obtain a~~ wherein said transforming the first filter into the second simplified filter ~~F2, it consists of~~ comprises:

- transforming the first ~~complex~~ filter into a combination of conditions on the attributes joined by the logical operators HAND, OR and NOT, the first filter being a complex filter;

- pushing the NOT operators to ~~the leaves~~to outer portions of a tree representation of the second simplified filter, and deleting ~~the~~ occurrences of double NOTs (NOT NOT);
- deleting ~~the~~ conditions ~~X~~ affecting the attributes that are not indexes;
- simplifying the resulting operations by replacing operands;
- factoring the nested ANDs and ORs;
- gathering the conditions related to the same index; and
- gathering all the ORs at the route of the filter and simplifying again.

7. (currently amended) ~~A~~ The method according to claim 6, ~~characterized in that in order to delete~~ in which said deleting the conditions ~~X, it consists of~~affecting the attributes that are not indexes comprisies replacing the conditions X and NOT X with the constant TRUE.

8. (currently amended) ~~A~~ The method according to claim 6, ~~characterized in that in order to~~ in which said simplifying the resulting operations~~, it consists of~~comprises:

- replacing ~~the~~ AND and OR tests ~~having~~ associated with only one operand with ~~this~~ the one operand;
- replacing ~~the~~ AND operations containing only TRUE operands with the constant TRUE, and replacing ~~the~~ OR operations containing only FALSE operands with the constant FALSE;
- removing ~~the~~ TRUE conditions from ~~the~~ other AND operations, and ~~the~~ removing FALSE conditions from the other OR operations;

- replacing ~~the~~ OR operations containing at least one TRUE operation with the constant TRUE, and ~~the~~ replacing AND operations containing at least one FALSE operand with the constant FALSE;

- replacing ~~the~~ conditions that are always TRUE or FALSE with the constant TRUE or FALSE; and

- ~~all of these simplification operations being applied as many times as it is possible to do so~~repeating said simplifying the resulting operations until no further simplifying is possible.

9. (currently amended) ~~A~~ The method according to claim 2, ~~characterized in that, in the second step, it consists~~in which said determining a first potential instance that verifies the second filter comprises:

~~of~~ concatenating ~~the first~~ a value that verifies $C1_{(i)}$ with ~~the~~ a first value that verifies $C2_{(i)}$, and so on up to $Cn_{(i)}$, in order to obtain ~~the~~ one or more zero local potential instances $I1\_0_{(i)}.I2\_0_{(i)}. \ldots In\_0_{(i)}$; and

selecting as the first potential instance the first possible value without a condition on a given index being the null value, the potential instance corresponding to the smallest of the zero local potential instances.

10. (currently amended) ~~A~~ The method according to claim 9, ~~characterized in that, in the fifth step, it~~further comprising:

~~consists of~~ performing, for any i and as long as ~~the~~ an index p is greater than 0, or as long as no instance searched for has been found, the following operations[[:]]

if there exists a $Jp_{(i)} > Ip$ that verifies the condition $Cp_{(i)}$, then the local potential instance is formed ~~in the following way:~~by

- for any index $k < p$, ~~we~~ take the value Ik with I1.I2. … .In being the identifier of the solution instance;
- for the index p, ~~we~~ take the value $Jp_{(i)}$; and
- for any index $k > p$, ~~we~~ take the value $Ik\_0_{(i)}$;

Otherwise, p takes the value p-1 and the method repeats the above operations, the potential instance corresponding to the smallest of the local potential instances obtained.

11. (currently amended) ~~A~~ The method according to claim 2, ~~characterized in that in the second and fifth steps, it consists of~~in which obtaining the test identifier from the identifier of the potential instance~~,~~ is performed by subtracting one from ~~its~~ a last number of the test identifier if the identifier of the potential instance~~latter~~ is different from 0, or by deleting ~~this~~ the last number if ~~it~~ the last number is null.

12. (currently amended) A system for processing complex requests and for optimizing the number of the SNMP requests transmitted through a network, ~~processing a~~the complex request addressed to at least one SNMP agent ~~(5)~~ of a resource machine ~~(2b)~~ of a computer system ~~(1)~~ from a complex protocol manager ~~(4)~~ of an application machine ~~(2a)~~, each said agent ~~(5)~~being configured to ~~managing~~ manage one or more attribute tables ~~belonging to~~associated with the resource machine ~~(2b)~~, in which ~~the~~ instances of the tables ~~being~~ are referenced by identifiers comprising indexes, the system comprising:

-an integrating agent ~~(6)~~ configured to transform a first filter derived from a complex request from the manager of the application machine into a second simplified filter comprising only conditions on indexes, the second simplified filter corresponding to the following matching characteristics: the second simplified filter lets through all the SNMP requests whose responses could verify the first filter, based on conditions whose attribute values could verify the first filter, and the second simplified filter filters out all the SNMP requests whose responses cannot in any way verify the first filter because the conditions on indexes associated with said filtered-out SNMP requests do not verify the first filter regardless of attribute values associated with said conditions;

wherein the transforming further comprises deleting from the first filter all conditions that operate on attributes that are not associated with any of said indexes; and

the integrating agent further configured to limit the SNMP requests to those that comply with the second simplified filter, to transmit said limited SNMP requests to the SNMP agent of the resource machine through the network, and to apply the first filter to the responses obtained to the SNMP requests.

~~that makes it possible to implement the processing method according to claim 1.~~

13. (currently amended) A method for processing a complex request and to optimize the number of the SNMP requests transmitted through a network, wherein the complex request is addressed to at least one SNMP agent (5) of a resource machine (2b) of a computer system (1) from a complex protocol manager (4) of an

application machine (2a), wherein the complex request addressed to the agent (5) from the manager (4) comprises SNMP attributes managed by the agent (5) and capable of being represented by a filter (F1, F2) constituted by any number of conditions on any number of attributes, linked to one another by any number of Boolean operators (AND, OR, NOT, EX.OR, etc.) and the application (2a) and resource (2b) machines communicate through a network (3), each agent (5) managing attribute tables belonging to the resource machine (2b), the instances of the tables being referenced by identifiers comprising indexes, comprising:

- transforming a complex filter (F1) derived from the complex request addressed to agent (5) from the manager (4) of the application machine (2a) into a simplified filter (F2) comprising only conditions on indexes, ~~and~~ the simplified filter (F2) adapted to let through all the SNMP requests whose responses could verify the complex filter (F1), based on conditions whose attribute values could verify the complex filter (F1), and to ~~but~~ filter out all the SNMP requests whose responses cannot in any way verify the complex filter (F1) because the conditions on indexes associated with said filtered-out SNMP requests do not verify the first filter (F1) regardless of attribute values associated with said conditions;

wherein the transforming further comprises deleting from the complex filter (F1) all conditions that operate on attributes that are not associated with any of said indexes;

- limiting the SNMP requests to those that comply with the simplified filter (F2);

- transmitting said limited SNMP requests to the SNMP agent (5) of the resource machine (2b) through the network (3); and

~~•~~—applying the complex filter (F1) to the responses obtained to the SNMP requests~~;~~

~~to thereby process said complex request and to optimize the number of the SNMP requests transmitted through the network (3)~~.

14. (currently amended) A method according to claim 13, wherein an identifier just below an identifier of the potential instance determined is a test identifier, the method further comprising:

1) determining ~~the~~ a first potential instance that verifies the simplified filter (F2);

2) using an SNMP request to find ~~the~~ an instance of the table having as its identifier ~~the one~~an identifier that follows ~~the~~ a test identifier and if no instance of the table is found, terminating ~~the~~ processing of the method, and if an instance is found, naming the instance found- a solution instance;

3) determining whether the solution instance is part of the response to the complex request processed by verifying the complex filter (F1) and upon verification of the complex filter (F1), applying the complex filter (F1) to the solution instance; and

4) determining the first potential instance whose identifier is higher than the identifier of the solution instance and that verifies the simplified filter (F2) and terminating ~~the~~ processing of the method if no instance is found, and if an instance is found, naming the identifier that is just below the identifier of the potential instance a test identifier and resuming the step

of using the SNMP request to find the instance of the table having as its identifier the ~~one~~ identifier that follows the test identifier.[[.]]

15. (previously presented) A method according to claim 14 comprising in the step of transforming the complex filter (F1) into the simplified filter (F2) having the form:

(OR

(AND

condition on index 1: $C1_{(1)}$

condition on index 2: $C2_{(1)}$

...

condition on index n: $Cn_{(1)}$

)

...

(AND

condition on index 1: $C1_{(i)}$

condition on index 2: $C2_{(i)}$

...

condition on index n: $Cn_{(i)}$

)

...

).

16. (currently amended) A method according to claim 14, wherein in the first step, after simplification, the simplified filter (F2) is reduced to:

- only the TRUE condition, in which case the table is scanned in its entirety; and
- only the FALSE condition, in which case no instance can work.

17. (currently amended) A method according to claim 15, wherein in the first step, after simplification, the simplified filter (F2) is reduced to:

- only the TRUE condition, in which case the table is scanned in its entirety; and
- only the FALSE condition, in which case no instance can work.

18. (currently amended) ~~A~~ The method according to claim 14, ~~characterized in that,~~further comprising: in order to obtain the simplified filter F2, immediately verifying whether the complex filter responds to rules defining filters that are not verified by any instance.

19. (currently amended) ~~A~~ The method according to claim 15, ~~characterized in that,~~further comprising: in order to obtain the simplified filter F2, immediately verifying whether the complex filter responds to rules defining filters that are not verified by any instance.

20. (currently amended) ~~A~~ The method according to claim 16, ~~characterized in that,~~further comprising: in order to obtain the simplified filter F2, immediately

verifying whether the complex filter responds to rules defining filters that are not verified by any instance.

21. (currently amended) ~~A~~ The method according to claim 17, ~~characterized in that,~~ further comprising: in order to obtain the simplified filter F2, immediately verifying whether the complex filter responds to rules defining filters that are not verified by any instance.

22. (currently amended) ~~A~~ The method according to claim 13, further comprising: ~~characterized in that,~~

in order to obtain a simplified filter F2,

- transforming the complex filter (F1) into a combination of conditions on the attributes joined by the logical operators AND, OR and NOT;
- pushing NOT operators to the leaves of a tree representing the simplified filter and deleting double NOTs (NOT NOT);
- deleting the conditions X affecting the attributes that are not indexes;
- simplifying the resulting operations;
- factoring the nested ANDs and ORs;
- gathering the conditions related to the same index; and
- gathering all the ORs at the route of the filter and simplifying the resulting operations again.

23. (currently amended) ~~A~~ The method according to claim 14, further comprising: ~~characterized in that,~~

in order to obtain a simplified filter F2,

- transforming the complex filter (F1) into a combination of conditions on the attributes joined by the logical operators AND, OR and NOT;
- pushing NOT operators to the leaves of a tree representation of the simplified filter and deleting double NOTs (NOT NOT);
- deleting the conditions X affecting the attributes that are not indexes;
- simplifying the resulting operations;
- factoring the nested ANDs and ORs;
- gathering the conditions related to the same index; and
- gathering all the ORs at the route of the filter and simplifying the resulting operations again.

24. (currently amended) A method according to claim 15, further comprising: ~~characterized in that,~~

in order to obtain a simplified filter F2,

- transforming the complex filter (F1) into a combination of conditions on the attributes joined by the logical operators AND, OR and NOT;
- pushing NOT operators to the leaves of a tree representation of the simplified filter and deleting double NOTs (NOT NOT);
- deleting the conditions X affecting the attributes that are not indexes;
- simplifying the resulting operations;
- factoring the nested ANDs and ORs;
- gathering the conditions related to the same index; and

- gathering all the ORs at the route of the filter and simplifying the resulting operations again.

25. (currently amended) ~~A~~ The method according to claim 16, further comprising: ~~characterized in that,~~

in order to obtain a simplified filter F2,

- transforming the complex filter (F1) into a combination of conditions on the attributes joined by the logical operators AND, OR and NOT;
- pushing NOT operators to the leaves of a tree representation of the simplified filter and deleting double NOTs (NOT NOT);
- deleting the conditions X affecting the attributes that are not indexes;
- simplifying the resulting operations;
- factoring the nested ANDs and ORs;
- gathering the conditions related to the same index; and
- gathering all the ORs at the route of the filter and simplifying the resulting operations again.

26. (currently amended) A method according to claim 17, further comprising: ~~characterized in that,~~

in order to obtain a simplified filter F2,

- transforming the complex filter (F1) into a combination of conditions on the attributes joined by the logical operators AND, OR and NOT;
- pushing NOT operators to the leaves of a tree representation of the simplified filter and deleting double NOTs (NOT NOT);

- deleting the conditions X affecting the attributes that are not indexes;
- simplifying the resulting operations;
- factoring the nested ANDs and ORs;
- gathering the conditions related to the same index; and
- gathering all the ORs at the route of the filter and simplifying the resulting operations again.

27. (currently amended) A method according to claim 18, further comprising: ~~characterized in that,~~

in order to obtain a simplified filter F2,

- transforming the complex filter (F1) into a combination of conditions on the attributes joined by the logical operators AND, OR and NOT;
- pushing NOT operators to the leaves of a tree representation of the simplified filter and deleting double NOTs (NOT NOT);
- deleting the conditions X affecting the attributes that are not indexes;
- simplifying the resulting operations;
- factoring the nested ANDs and ORs;
- gathering the conditions related to the same index; and
- gathering all the ORs at the route of the filter and simplifying the resulting operations again.

28. (previously presented) A method according to claim 22, comprising replacing the conditions X and NOT X with the constant TRUE in order to delete the conditions X.

29. (previously presented) A method according to claim 23, comprising replacing the conditions X and NOT X with the constant TRUE in order to delete the conditions X.

30. (previously presented) A method according to claim 24, comprising replacing the conditions X and NOT X with the constant TRUE in order to delete the conditions X.

31. (previously presented) A method according to claim 25, comprising replacing the conditions X and NOT X with the constant TRUE in order to delete the conditions X.

32. (currently amended) ~~A~~ The method according to claim 18, having AND and OR operations and ~~characterized in that in order to simplify the~~comprising further simplifying operations, ~~it consists of~~the method comprising:

- replacing AND and OR operations having only one operand with said one operand;
- replacing AND operations containing only TRUE operands with a constant TRUE, and replacing OR operations containing only FALSE operands with a constant FALSE;
- removing TRUE conditions from the other AND operations, and removing FALSE conditions from the other OR operations;

- replacing OR operations containing at least one TRUE operation with a constant TRUE, and replacing AND operations containing at least one FALSE operand with a constant FALSE;

- replacing conditions that are always TRUE with a constant TRUE, and replacing conditions that are always FALSE with a constant FALSE; and

~~all of said latter simplification~~repeating each said further simplifying operation~~s being repeated~~ as many times as it is possible to do so.

33. (currently amended) ~~A~~ The method according to claim 23, having AND and OR operations and ~~characterized in that in order to simplify the~~comprising further simplifying operations, ~~it consists of~~the method comprising:

- replacing AND and OR operations having only one operand with said one operand;

- replacing AND operations containing only TRUE operands with a constant TRUE, and replacing OR operations containing only FALSE operands with a constant FALSE;

- removing TRUE conditions from the other AND operations, and removing FALSE conditions from the other OR operations;

- replacing OR operations containing at least one TRUE operation with a constant TRUE, and replacing AND operations containing at least one FALSE operand with a constant FALSE;

- replacing conditions that are always TRUE with a constant TRUE, and replacing conditions that are always FALSE with a constant FALSE; and

~~all of said latter simplification~~ repeating each said further simplifying operations~~ being repeated~~ as many times as it is possible to do so.

34. (currently amended) ~~A~~ The method according to claim 24, having AND and OR operations and ~~characterized in that in order to simplify the~~comprising further simplifying operations, ~~it consists of~~the method comprising:

- replacing AND and OR operations having only one operand with said one operand;
- replacing AND operations containing only TRUE operands with a constant TRUE, and replacing OR operations containing only FALSE operands with a constant FALSE;
- removing TRUE conditions from the other AND operations, and removing FALSE conditions from the other OR operations;
- replacing OR operations containing at least one TRUE operation with a constant TRUE, and replacing AND operations containing at least one FALSE operand with a constant FALSE;
- replacing conditions that are always TRUE with a constant TRUE, and replacing conditions that are always FALSE with a constant FALSE; and

~~all of said latter simplification~~repeating each said further simplifying operations~~ being repeated~~ as many times as it is possible to do so.

35. (currently amended) ~~A~~ The method according to claim 25, having AND and OR operations and ~~characterized in that in order to simplify the~~comprising further simplifying operations, ~~it consists of~~the method comprising:

- replacing AND and OR operations having only one operand with said one operand;

- replacing AND operations containing only TRUE operands with a constant TRUE, and replacing OR operations containing only FALSE operands with a constant FALSE;

- removing TRUE conditions from the other AND operations, and removing FALSE conditions from the other OR operations;

- replacing OR operations containing at least one TRUE operation with a constant TRUE, and replacing AND operations containing at least one FALSE operand with a constant FALSE;

- replacing conditions that are always TRUE with a constant TRUE, and replacing conditions that are always FALSE with a constant FALSE; and

~~all of said latter simplification~~repeating each said further simplifying operation~~s being repeated~~ as many times as it is possible to do so.

36. (currently amended) ~~A~~ The method according to claim 26, having AND and OR operations and ~~characterized in that in order to simplify the~~comprising further simplifying operations, ~~it consists of~~the method comprising:

- replacing AND and OR operations having only one operand with said one operand;

- replacing AND operations containing only TRUE operands with a constant TRUE, and replacing OR operations containing only FALSE operands with a constant FALSE;

- removing TRUE conditions from the other AND operations, and removing FALSE conditions from the other OR operations;
- replacing OR operations containing at least one TRUE operation with a constant TRUE, and replacing AND operations containing at least one FALSE operand with a constant FALSE;
- replacing conditions that are always TRUE with a constant TRUE, and replacing conditions that are always FALSE with a constant FALSE; and

~~all of said latter simplification~~repeating each said further simplifying operation~~s being repeated~~ as many times as it is possible to do so.

37. (currently amended) ~~A~~ The method according to claim 27, having AND and OR operations and ~~characterized in that in order to simplify the~~comprising further simplifying operations, ~~it consists of~~the method comprising:

- replacing AND and OR operations having only one operand with said one operand;
- replacing AND operations containing only TRUE operands with a constant TRUE, and replacing OR operations containing only FALSE operands with a constant FALSE;
- removing TRUE conditions from the other AND operations, and removing FALSE conditions from the other OR operations;
- replacing OR operations containing at least one TRUE operation with a constant TRUE, and replacing AND operations containing at least one FALSE operand with a constant FALSE;

- replacing conditions that are always TRUE with a constant TRUE, and

replacing conditions that are always FALSE with a constant FALSE; and

~~all of said latter simplification~~repeating each said further simplifying operations ~~being repeated~~ as many times as it is possible to do so.

38. (currently amended) ~~A~~ The method according to claim 28, having AND and OR operations and ~~characterized in that in order to simplify the~~comprising further simplifying operations, ~~it consists of~~the method comprising:

- replacing AND and OR operations having only one operand with said one operand;
- replacing AND operations containing only TRUE operands with a constant TRUE, and replacing OR operations containing only FALSE operands with a constant FALSE;
- removing TRUE conditions from the other AND operations, and removing FALSE conditions from the other OR operations;
- replacing OR operations containing at least one TRUE operation with a constant TRUE, and replacing AND operations containing at least one FALSE operand with a constant FALSE;
- replacing conditions that are always TRUE with a constant TRUE ,and

replacing conditions that are always FALSE with a constant FALSE; and

~~all of said latter simplification~~repeating each said further simplifying operations ~~being repeated~~ as many times as it is possible to do so.

39. (currently amended) ~~A~~ The method according to claim 29, having AND and OR operations and ~~characterized in that in order to simplify the~~comprising further simplifying operations, ~~it consists of~~the method comprising:

- replacing AND and OR operations having only one operand with said one operand;
- replacing AND operations containing only TRUE operands with a constant TRUE, and replacing OR operations containing only FALSE operands with a constant FALSE;
- removing TRUE conditions from the other AND operations, and removing FALSE conditions from the other OR operations;
- replacing OR operations containing at least one TRUE operation with a constant TRUE, and replacing AND operations containing at least one FALSE operand with a constant FALSE;
- replacing conditions that are always TRUE with a constant TRUE, and replacing conditions that are always FALSE with a constant FALSE; and

~~all of said latter simplification~~repeating each said further simplifying operations ~~being repeated~~ as many times as it is possible to do so.

40. (currently amended) ~~A~~ The method according to claim 30, having AND and OR operations and ~~characterized in that in order to simplify the~~comprising further simplifying operations, ~~it consists of~~the method comprising:

- replacing AND and OR operations having only one operand with said one operand;

- replacing AND operations containing only TRUE operands with a constant TRUE, and replacing OR operations containing only FALSE operands with a constant FALSE;

- removing TRUE conditions from the other AND operations, and removing FALSE conditions from the other OR operations;

- replacing OR operations containing at least one TRUE operation with a constant TRUE, and replacing AND operations containing at least one FALSE operand with a constant FALSE;

- replacing conditions that are always TRUE with a constant TRUE, and replacing conditions that are always FALSE with a constant FALSE; and

~~all of said latter simplification~~repeating each said further simplifying operation~~s being repeated~~ as many times as it is possible to do so.

41. (currently amended) ~~A~~ The method according to claim 31, having AND and OR operations and ~~characterized in that in order to simplify the~~comprising further simplifying operations, ~~it consists of~~the method comprising:

- replacing AND and OR operations having only one operand with said one operand;

- replacing AND operations containing only TRUE operands with a constant TRUE, and replacing OR operations containing only FALSE operands with a constant FALSE;

- removing TRUE conditions from the other AND operations, and removing FALSE conditions from the other OR operations;

- replacing OR operations containing at least one TRUE operation with a constant TRUE, and replacing AND operations containing at least one FALSE operand with a constant FALSE;
- replacing conditions that are always TRUE with a constant TRUE, and replacing conditions that are always FALSE with a constant FALSE; and

~~all of said latter simplification~~repeating each said further simplifying operation~~s being repeated~~ as many times as it is possible to do so.

42. (currently amended) ~~A~~ The method according to claim 14, ~~characterized in that~~wherein the step of determining the first potential instance that verifies the simplified filter comprises concatenating the first value that verifies $C1_{(i)}$ with the first value that verifies $C2_{(i)}$, and so on up to $Cn_{(i)}$, in order to obtain zero local potential instances $I1\_0_{(i)}.I2\_0_{(i)}. \ldots In\_0_{(i)}$, the first possible value without a condition on a given index being the null value, the potential instance corresponding to the smallest of the zero local potential instances.

43. (currently amended) ~~A~~ The method according to claim 15, ~~characterized in that~~ wherein the step of determining the first potential instance that verifies the simplified filter comprises concatenating the first value that verifies $C1_{(i)}$ with the first value that verifies $C2_{(i)}$, and so on up to $Cn_{(i)}$, in order to obtain zero local potential instances $I1\_0_{(i)}.I2\_0_{(i)}. \ldots In\_0_{(i)}$, the first possible value without a condition on a given index being the null value, the potential instance corresponding to the smallest of the zero local potential instances.

44. (currently amended) ~~A~~ The method according to claim 16, ~~characterized in that~~ wherein the step of determining the first potential instance that verifies the simplified filter comprises concatenating the first value that verifies $C1_{(i)}$ with the first value that verifies $C2_{(i)}$, and so on up to $Cn_{(i)}$, in order to obtain zero local potential instances $I1\_0_{(i)}.I2\_0_{(i)}. \ldots In\_0_{(i)}$, the first possible value without a condition on a given index being the null value, the potential instance corresponding to the smallest of the zero local potential instances.

45. (currently amended) ~~A~~ The method according to claim 18, ~~characterized in that~~ wherein the step of determining the first potential instance that verifies the simplified filter comprises concatenating the first value that verifies $C1_{(i)}$ with the first value that verifies $C2_{(i)}$, and so on up to $Cn_{(i)}$, in order to obtain zero local potential instances $I1\_0_{(i)}.I2\_0_{(i)}. \ldots In\_0_{(i)}$, the first possible value without a condition on a given index being the null value, the potential instance corresponding to the smallest of the zero local potential instances.

46. (currently amended) ~~A~~ The method according to claim 22, ~~characterized in that~~ wherein the step of determining the first potential instance that verifies the simplified filter comprises concatenating the first value that verifies $C1_{(i)}$ with the first value that verifies $C2_{(i)}$, and so on up to $Cn_{(i)}$, in order to obtain zero local potential instances $I1\_0_{(i)}.I2\_0_{(i)}. \ldots In\_0_{(i)}$, the first possible value without a condition on a given index being the null value, the potential instance corresponding to the smallest of the zero local potential instances.

47. (currently amended) ~~A~~ The method according to claim 28, ~~characterized in that~~ wherein the step of determining the first potential instance that verifies the simplified filter comprises concatenating the first value that verifies $C1_{(i)}$ with the first value that verifies $C2_{(i)}$, and so on up to $Cn_{(i)}$, in order to obtain zero local potential instances $I1\_0_{(i)}.I2\_0_{(i)}$. ... $In\_0_{(i)}$, the first possible value without a condition on a given index being the null value, the potential instance corresponding to the smallest of the zero local potential instances.

48. (currently amended) ~~A~~ The method according to claim 32, ~~characterized in that~~ wherein the step of determining the first potential instance that verifies the simplified filter comprises concatenating the first value that verifies $C1_{(i)}$ with the first value that verifies $C2_{(i)}$, and so on up to $Cn_{(i)}$, in order to obtain zero local potential instances $I1\_0_{(i)}.I2\_0_{(i)}$. ... $In\_0_{(i)}$, the first possible value without a condition on a given index being the null value, the potential instance corresponding to the smallest of the zero local potential instances.

49. (currently amended) ~~A~~ The method according to claim 42, ~~characterized in that~~ wherein the step of determining the first potential instance whose identifier is higher than the identifier of the solution instance comprises performing, for any i and as long as the index p is greater than 0 or as long as no instance searched for has been found, the following operations:

if there exists a $Jp_{(i)} > Ip$ that verifies the condition $Cp_{(i)}$, then the local potential instance is formed in the following way:

- for any index k < p, we take the value Ik with I1.I2. … .In being the identifier of the solution instance;
- for the index p, we take the value $Jp_{(i)}$; and
- for any index k > p, we take the value $Ik\_0_{(i)}$;

otherwise p takes the value p-1 and the method repeats the above operations, the potential instance corresponding to the smallest of the local potential instances obtained.

50. (currently amended) ~~A~~ The method according to claim 43, ~~characterized in that~~ wherein the step of determining the first potential instance whose identifier is higher than the identifier of the solution instance comprises performing, for any i and as long as the index p is greater than 0 or as long as no instance searched for has been found, the following operations:

if there exists a $Jp_{(i)} > Ip$ that verifies the condition $Cp_{(i)}$, then the local potential instance is formed in the following way:

- for any index k < p, we take the value Ik with I1.I2. … .In being the identifier of the solution instance;
- for the index p, we take the value $Jp_{(i)}$; and
- for any index k > p, we take the value $Ik\_0_{(i)}$;

otherwise p takes the value p-1 and the method repeats the above operations, the potential instance corresponding to the smallest of the local potential instances obtained.

51. (currently amended) ~~A~~ The method according to claim 44, ~~characterized in that~~ wherein the step of determining the first potential instance whose identifier is higher than the identifier of the solution instance comprises performing, for any i and as long as the index p is greater than 0 or as long as no instance searched for has been found, the following operations:

if there exists a $Jp_{(i)} > Ip$ that verifies the condition $Cp_{(i)}$, then the local potential instance is formed in the following way:

- for any index $k < p$, we take the value Ik with I1.I2. ... .In being the identifier of the solution instance;
- for the index p, we take the value $Jp_{(i)}$; and
- for any index $k > p$, we take the value $Ik\_0_{(i)}$;

otherwise p takes the value p-1 and the method repeats the above operations, the potential instance corresponding to the smallest of the local potential instances obtained.

52. (currently amended) ~~A~~ The method according to claim 45, ~~characterized in that~~ wherein the step of determining the first potential instance whose identifier is higher than the identifier of the solution instance comprises performing, for any i and as long as the index p is greater than 0 or as long as no instance searched for has been found, the following operations:

if there exists a $Jp_{(i)} > Ip$ that verifies the condition $Cp_{(i)}$, then the local potential instance is formed in the following way:

- for any index $k < p$, we take the value Ik with I1.I2. ... .In being the identifier of the solution instance;

- for the index p, we take the value $Jp_{(i)}$; and
- for any index k > p, we take the value $Ik\_0_{(i)}$;

otherwise p takes the value p-1 and the method repeats the above operations, the potential instance corresponding to the smallest of the local potential instances obtained.

53. (currently amended) ~~A~~ The method according to claim 14, ~~characterized in that~~wherein the steps of determining the first potential instance that verifies the simplified filter and the first potential instance whose identifier is higher than the identifier of the solution instance consist of obtaining the test identifier from the identifier of the potential instance, by subtracting one from its last number if the latter is different from 0, or by deleting this last number if it is null.

54. (currently amended) ~~A~~ The method according to claim 15, ~~characterized in that~~wherein the steps of determining the first potential instance that verifies the simplified filter and the first potential instance whose identifier is higher than the identifier of the solution instance consist of obtaining the test identifier from the identifier of the potential instance, by subtracting one from its last number if the latter is different from 0, or by deleting this last number if it is null.

55. (currently amended) ~~A~~ The method according to claim 16, ~~characterized in that~~wherein the steps of determining the first potential instance that verifies the simplified filter and the first potential instance whose identifier is higher than the identifier of the solution instance consist of obtaining the test identifier from the

identifier of the potential instance, by subtracting one from its last number if the latter is different from 0, or by deleting this last number if it is null.

56. (currently amended) ~~A~~ The method according to claim 18, ~~characterized in that~~wherein the steps of determining the first potential instance that verifies the simplified filter and the first potential instance whose identifier is higher than the identifier of the solution instance consist of obtaining the test identifier from the identifier of the potential instance, by subtracting one from its last number if the latter is different from 0, or by deleting this last number if it is null.

57. (currently amended) ~~A~~ The method according to claim 22, ~~characterized in that~~wherein the steps of determining the first potential instance that verifies the simplified filter and the first potential instance whose identifier is higher than the identifier of the solution instance consist of obtaining the test identifier from the identifier of the potential instance, by subtracting one from its last number if the latter is different from 0, or by deleting this last number if it is null.

58. (currently amended) ~~A~~ The method according to claim 28, ~~characterized in that~~wherein the steps of determining the first potential instance that verifies the simplified filter and the first potential instance whose identifier is higher than the identifier of the solution instance consist of obtaining the test identifier from the identifier of the potential instance, by subtracting one from its last number if the latter is different from 0, or by deleting this last number if it is null.

59. (currently amended) ~~A~~ The method according to claim 32, ~~characterized in that~~ wherein the steps of determining the first potential instance that verifies the simplified filter and the first potential instance whose identifier is higher than the identifier of the solution instance consist of obtaining the test identifier from the identifier of the potential instance, by subtracting one from its last number if the latter is different from 0, or by deleting this last number if it is null.

60. (currently amended) ~~A~~ The method according to claim 42, ~~characterized in that~~ wherein the steps of determining the first potential instance that verifies the simplified filter and the first potential instance whose identifier is higher than the identifier of the solution instance consist of obtaining the test identifier from the identifier of the potential instance, by subtracting one from its last number if the latter is different from 0, or by deleting this last number if it is null.

61. (currently amended) ~~A~~ The method according to claim 49, ~~characterized in that~~ wherein the steps of determining the first potential instance that verifies the simplified filter and the first potential instance whose identifier is higher than the identifier of the solution instance consist of obtaining the test identifier from the identifier of the potential instance, by subtracting one from its last number if the latter is different from 0, or by deleting this last number if it is null.

62. (currently amended) A system for processing a complex request and to optimize the number of the SNMP requests transmitted through a network, ~~complex request~~ comprising at least one SNMP agent (5) of a resource machine (2b) of a

computer system (1) to which the complex request is transmitted from a complex protocol manager (4) of an application machine (2a), each agent (5) managing attribute tables belonging to the resource machine (2b), instances of the tables being referenced by identifiers comprising indexes, the system comprising an integrating agent (6) for processing the complex request,

means for transforming a complex filter (F1) derived from the complex request addressed to agent (5) from the manager (4) of the application machine (2a) into a simplified filter (F2) comprising only conditions on indexes, the ~~complex~~ simplified filter (F2) adapted to let through all SNMP requests whose responses could verify the ~~simplified~~ complex filter (F1), based on conditions whose attribute values could verify the complex filter (F1), and to ~~but~~ filter out all SNMP requests whose responses cannot in any way verify the ~~simplified~~ complex filter (F1) because the conditions on indexes associated with said filtered-out requests do not verify the complex filter (F1) regardless of attribute values associated with said conditions;

wherein the transforming further comprises deleting from the complex filter (F1) all conditions that operate on attributes that are not associated with any of said indexes;

means for limiting SNMP requests to those that comply with the complex filter (F2);

means for transmitting said limited SNMP requests to the SNMP agent (5) of the resource machine (2b) through the network (3); and

means for applying the simplified filter (F1) to the responses obtained to the SNMP requests~~;~~

~~to thereby process said complex request and to optimize the number of the SNMP requests transmitted through the network (3)~~.

63. (previously presented) The system for processing as set forth in claim 62 further comprising means for determining the first potential instance that verifies the simplified filter (F2) wherein the identifier first below the identifier of the potential instance determined is a test identifier.

64. (currently amended) The system for processing as set forth in claim 63 wherein, using an SNMP request, there is provided means to find the instance of the table having as its identifier the one that follows the test identifier and if no instance is found, terminating the processing method, if an instance is found, naming the instance found a solution instance; and means for determining whether the solution instance is part of the response to the complex request processed by verifying the complex filter (F1) and upon verification of the complex ~~Sfilter~~ filter (F1), applying the complex filter (F1) to the solution instance[[;]].

65. (previously presented) The system for processing as set forth in claim 64 further comprising means for transforming the complex filter (F1) into a simplified filter having the form

(OR

(AND

condition on index 1: $C1_{(1)}$

condition on index 2: $C2_{(1)}$

...

condition on index n: $Cn_{(1)}$

)

...

(AND

condition on index 1: $C1_{(i)}$

condition on index 2: $C2_{(i)}$

...

condition on index n: $Cn_{(i)}$

)

...

).